Cascade Error Projection: An Efficient Hardware Learning Algorithm

Tuan A. Duong
Center for Space Microelectronics Technology
Jet Propulsion Laboratory, California Institute of Technology
Pasadena, CA91109
and
Department of Electrical Engineering, University of California, Irvine
h-vine, CA 92717
tuan@brain.jpl.nasa.gov

ABSTRACT

A new learning algorithm termed Cascade Error Projection (CEP) which provides efficient learning in hardware is presented. This algorithm is adapted a constructive architecture from cascade correlation and the dynamical stepsize of AID conversion from the cascade back propagation algorithm. The CEP technique is faster to execute, because part of the weights are deterministically obtained, *and the learning of the remaining weights from the inputs to the hidden unit is performed as a single-layer perceptron learning with previously determined weights kept frozen. In addition, one can start out with zero weight values (rather than random finite weight values) when the learning of each layer is commenced. Further, unlike cascade correlation algorithm (where a pool of candidate hidden units is added), only a single hidden unit is added at a time. Therefore, the* simplicity *in hardware implementation is also achieved. In simulation, a fixed* 100 *epoch iterations is used for each single-layer perception (each single hidden unit) learning. The highlight of this algorithm is that with round-off method, 5-to 8-bit parity problems can be solved with a limited synaptic resolution of only 3- to 4-bit, and the same problem with truncation technique would be solved with 5-bit or nwre synaptic resolution.*

J. Introduction

There are many ill-defined problems in pattern recognition, classification, vision, and speech recognition which need to be solved in real time [1-4]. These problems are too complex to be solved by a linear technique; the most suitable approach would be a non-linear technique, such as a neural net work. Error Backpropagation (EBP)[5] learning algorithm is a popular supervised learning technique. Onc of the most attractive features of the neural network is a massively parallel processing that offers tremendous speed only when implemented in hardware. From the hardware point of view, EBP requires at least 12-bit weight resolution to learn a non-linear simple problem[6] showing that this learning algorithm is very expensive for hard ware implementation. However, there arc other learning algorithms that arc more practical than others [7]. There have been a fcw attempts to adapt such algorithms to hardware. Weight perturbation[8], cascade correlat ion(CC)[9], and cascade backpropagat ion (CBP)[1O] approaches has been investigated and reported in literature,

In this paper, a ncw efficient cascade error projection (CEP) learning algorithm is presented. It offers a

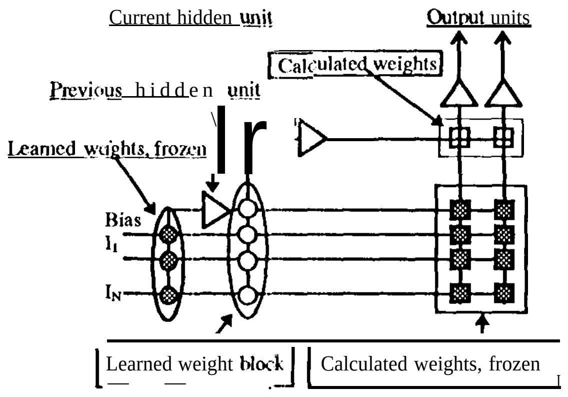

Figure 1. The architecture of cascade error projection includes inputs, hidden units, and output units. The shaded circles or squares indicate the learned or calculated weight set which are computed and frozen. A circle indicates that learning is applied to obtain the weight set using perceptron learning, and a square indicates that tbe weight set is deterministically calculated.

fast, reliable, and hardware implementable learning method using a one-layer perceptron approach and a deterministic calculation for the other layer. This procedure offers a fast, reliable, and hardware implementable learning algorithm [11]. To validate the new learning algorithm of CEP, simulations of problems from 5- to 8-bit parity were investigated with weight quantization based on a floating point machine (32-bit **for float and 64-bit** for double precision) as well as limited weight **quantizations based** on hard ware implementation (3- to 6-bit weight resolution) using two A/D conversion techniques (round-off and truncation).

In general, the network architecture is shown in Figure 1. Shaded squares and circles indicate frozen weights; a square indicates calculated weights, and a circle indicates learned weights.

2. Cascade Error Projection Learning Algorithm

The motivation to use the CEP technique is supported by three reasons:

a) It has faster learning by dividing the network into two sub-networks:
   i) Stochastic learning network.
   ii) Deterministic calculation network.
b) It is efficient even with limited weight resolution hence easier to implement in hard ware.
c) The analysis guarantees its learning capability both in continuous and in discrete weight space [11].

2.1 Mathematical approach:

The energy function is defined as:

$$E(n+1) = \sum_{p=1}^{P} \{f_h^p(n+1) - \frac{1}{m}\sum_{o=1}^{m}(t_o^p - o_o^p)\}^2$$

The weight updating between the inputs (including previously added hidden units) and the newly added hidden unit is calculated as follows:

$$\Delta w_{ih}^p(n+1) = -\eta \frac{\partial E(n+1)}{\partial w_{ih}^p(n+1)} \qquad \text{(a)}$$

and the weight updating between hidden unit $n+1$ and the output unit o is

$$w_{ho}(n+1) = \frac{\sum_{p=1}^{P} \varepsilon_o^p f'^p_o f_h^p(n+1)}{\sum_{p=1}^{P} [f'^p_o f_h^p(n+1)]^2}$$

$$\text{with } f(x) = \frac{1-e^{-x}}{1+e^{-x}} \qquad \text{(b)}$$

The used notations are defined as follows:
m is the number of outputs, P is the number of training patterns.

Error $\varepsilon_o^p = t_o^p - o_o^p(n)$; where $o_o^p(n)$ is the output element o of actual output $o(n)$ for training pattern p and t_o^p is the target element o for training pattern p. n indicates number of previously added hidden units.

y:(n) = f'^p_o denotes the output transfer function derivative with respect to its input.

$f_h^p(n+1)$ denotes the transfer function of hidden unit $n+1$.

3. Simulation

3.1 Cascade **Error Projection Learning Algorithm Procedure**

1. *Start with the network which has input and output neurons. With the given input and output patterns and hyperbolic transfer function, one can determine the set of weights between input and output by using pseudo-inverse or perceptron learning. The weight set W_{io} is thus obtained and frozen.*
2. *Add a new hidden unit with a zero weight set for each unit. In each loop (contains an epoch) an input-output pattern is picked up randomly in the epoch (no pattern repeated until every pattern in the epoch is picked). Use the perceptron learning technique of equation (a) 10 train $W_{ih}(n+1)$ for 100 epoch iterations.*
3. *stop the perceptron training. Calculate the weights $W_{ho}(n+1)$ between the current hidden unit and the output units from equation (b).*
4. *Cross-validate the network. If the criteria is satisfied, then stop training, and test the network. Otherwise, go to step 2 above until the number of hidden*

units is more than 20; then give up and quit!

3.2 Problems

From 5- to 8-bit parity problems arc solved in this paper. The input and output highs arc 0.8, and the lows arc -0.8. This simulation is conducted (1) with no limited weight quantization (32-bit for floating point or 64-bit for double precision); and, (2) the limited weight quantization from 3 to 6 bits using round-off and truncation techniques.

3.3 Parameters

The learning rate η is decreased linearly at each epoch as follows:

$$\eta_{new} = \eta_{old} - .01 * \eta_0$$

$\eta_0 = 0.4$ for 7- to 8-bit parity with no limited weight quantization, and $\eta_0 = 1.0$ for others.

3.4 Conversion techniques

The updating weight Aw is converted into the available weight quantization which is Aw*. The conversion can be summarized as follows:

$stepsize(n) = \beta E(n - 1)$ with β constant

*<u>Round-off technique</u>:

$$\Delta w^*_{jh}(n) = \begin{cases} stepsize(n) * \mathrm{int}(\frac{\Delta w_{jh}}{stepsize(n)} + 0.5) \\ if\ (\frac{w_{jh}(n)}{stepsize(n)} + \mathrm{int}(\frac{\Delta w_{jh}(n)}{stepsize(n)} + 0.5)) \\ \leq 2^B \quad and \quad \Delta w_{jh}(n) > O \\ \\ stepsize(n) * \mathrm{int}(\frac{\Delta w_{jh}(n)}{stepsize(n)} - 0.5) \\ if\ (\frac{w_{jh}(n)}{stepsize(n)} + \mathrm{int}(\frac{\Delta w_{jh}(n)}{stepsize(n)} - 0.5)) \\ \leq -2^B \quad and \quad \Delta w_{jh}(n) < O \\ \\ 0 \quad Otherwise \end{cases}$$

* <u>Truncation technique</u>

$$\Delta w^*_{jh}(n) = \begin{cases} stepsize(n) * \mathrm{int}(\frac{\Delta w_{jh}}{stepsize(n)}) \\ if\ (\frac{w_{jh}(n)}{stepsize(n)} + \mathrm{int}(\frac{\Delta w_{jh}(n)}{stepsize(n)})) \\ \leq 2^B \quad and \quad \Delta w_{jh}(n) > O \\ \\ stepsize(n) * \mathrm{int}(\frac{\Delta w_{jh}(n)}{stepsize(n)}) \\ if\ (\frac{w_{jh}(n)}{stepsize(n)} + \mathrm{int}(\frac{\Delta w_{jh}(n)}{stepsize(n)})) \\ \leq -2^B \quad and \quad \Delta w_{jh}(n) < 0 \\ \\ 0 \quad Otherwise \end{cases}$$

3.5 Simulation results

Figure 2 refers to simulation results with round-off technique. Even with 3-bit weight resolution the network is able to learn 5- to 7-bit parity problems with no error within the 20 hidden units limit. For weight quantization of 4-bit or more, the network reliably demonstrates the capability of learning from 5- to 8-bit parit y problems.

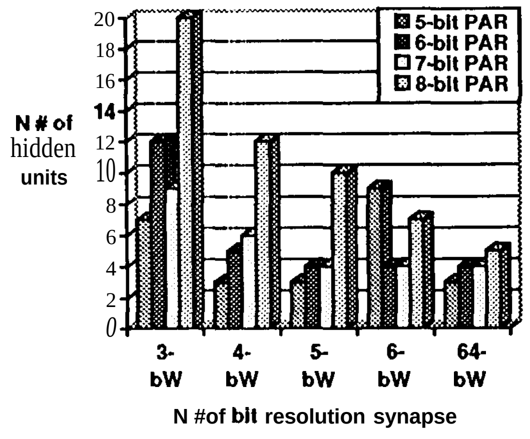

Figure 2: The chart shows CEP learning capability for 5- to S-bit parity problems using round-off technique. x axis represents limited weight quantization (3-6 and 64-bit) and y axis shows the resulting number of hidden units (limited to 20). Each hidden unit has 100 epoch iterations. As shown, the lager number of hidden units compensate for the lower weight resolution.

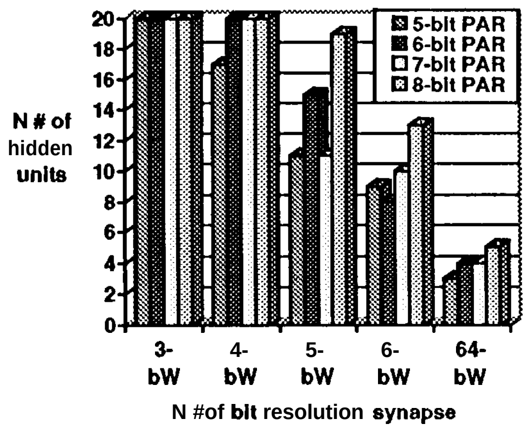

Figure 3: The chart shows CEP learning capability for 5- to 8-hit panty problems using truncation technique. x axis represents limited weight quantization (3-6 and 64-bit) and y axis shows the resulting number of hidden units (limited to 20). Each bidden unit has 100 epoch iterations. As shown, the lager number of hidden units compensate for the lower weight resolution.

Figure 3 shows similar results for the truncation technique. Here 3- and 4-bit weight quantizations do not have the capability to learn 5-bit or higher parity problems. However, with higher bit resolution the learning with truncation technique shows the learning capability; even though, it is not as consistent as that with the round-off technique.

4. Conclusions

in this paper, wc have shown that CEP is an efficient algorithm for hardware-based learning. Its advantages can be summarized as follows:

. A fast, reliable learning technique
- Easy to implement in hardware
- Tolerant of limited synaptic weight resolution
- The round-off technique is belter compared with truncation as shown by our analysis

5. Acknowledgment-s

The research described herein was performed by the Center for Space Microelectronics Technology, Jet Propulsion Laboratory, California Institute of Technology and Deparment of Electrical and Computer Engineering, University of California Irvine, and was jointly sponsored by the Ballistic Missile Dcfcnsc Orgrmization/Innovative Science and Technology Office (BMDO/IST), the Office of Naval Research (ONR), the Advanced Research projects Agency (ARPA), and the National Aeronautics and Space Administration (NASA). The author would like to thank Drs A. Stubberud, T. Daud, and A. Thakoor for useful discussions.

6. References

[1] B.E. Boser, E, Sackinger, J. Bromley, Y. LeCun, and L.D. Jackel, "An Analog Neural Network Processor with Programmable Topology," *IEEE Journal of Solid State Circuits, vol. 26, NO. 12, Dec. 1991.*

[2] T.A. Duong, T. Brown, M. Tran, H. Langenbacher, and T. Daud, "Analog VLSI neural network building block chips for hardware-in-the-loop learning," *Proc. IEEE/INNS Int'1 Join Conf. on Neural Networks,* Beijing, China, Nov. 3-6, 1992.

[3] T.A. Duong ct. al, "Low Power Analog Neurosynapse Chips for a 3-D "Sugarcube" Neuroprocessor," *Proc. of IEEE Intl' Conf. on Neural Networks*(ICNN/WCCI), Vol 111, pp. 1907-1911, June 28-July 2, 1994, Orlando, Florida.

[4] T.A. Duong ct. al., "Analog 3-D Neuroprocessor for Fast Frame Focal Plane Image Processing," *Journal of Simulation* , Vol 65, NO 1, pp. 11-25, July, 1995

[5] D.E. Rumelhart, and J.L. McClelland, *Parallel Distributed Processing: Explorations in the Microstructure of Cognition, Vol. 1: Foundation.* MIT Press, Cambridge, MA 1986.

[6] P. W. Hollis, J.S. Harper, and J.J. Paulos, "The effects of Precision Constraints in a Backpropagation learning Network," *Neural Computation, vol. 2,* pp. 363-373, 1990.

[7] R. Tawel, "Learning in analog neural network hard wat c," *Computers Elect. Engng*, Vol. 19, No. 6, pp 453-467, 1993.

[8] M. Jabri and B. Flower "Weight Perturbation: An Optimal Architecture and learning Technique for Analog VLSI Feedforward and recurrent Multilayer Networks," *IEEE Trans. on Neural Networks*, Vol 3, NO. 1, pp 154-157, Jan, 1992.

[9] M. Hochfeld and S. Fahlman, "Learning with limited numerical precision using the cascade-correlation algorithm ," *IEEE Trans. Neural Networks,* vol.3, No.4,pp602-611, July 1992.

[10] T.A. Duong, S.P. Eberhardt, T. Daud, and A. Thakoor, "Learning in neural networks: VLSI implementation strategies," In: Fuzzy logic and Neural Network Handbook, Ed: C.H. Chen, McGraw-Hill, 1995 (To be published).

[11] T.A. Duong, A. Stubberud, and T. Daud , "Cascade Error Projection learning theory" *submitted to Neural Computation*, 1995